Lab 2 – Thought Locker Prototype Product Specification

Olivia Gee

Old Dominion University

CS411W

Professor James Brunelle

22 October 2023

Lab2 Version 1
LAB 2 – THOUGHT LOCKER PROTOTYPE PRODUCT SPECIFICATION

Table of Contents

1 Introduction

   1.1 Purpose
   1.2 Scope
   1.3 Definitions, Acronyms, and Abbreviations
   1.4 References
   1.5 Overview

2 General Description

   2.1 Prototype Architecture Description
   2.2 Prototype Functional Description
   2.3 External Interfaces
      2.3.1 Hardware Interfaces
      2.3.2 Software Interfaces
      2.3.3 User Interfaces
      2.3.4 Communications Protocols and Interfaces

List of Figures

Figure 1: Thought Locker Prototype Major Functional Components

List of Tables

Table 1: Thought Locker Feature Description and Prototype Implementation
1 Introduction

In a world where dementia affects one in ten Americans aged 65 and older, our mobile assistant offers a lifeline to those with mild and moderate dementia. With features such as reminders, item location, monitoring, and analytics, Thought Locker empowers patients to maintain their independence, alleviating stress for caregivers. As dementia diagnoses steadily rise, Thought Locker stands as a vital tool for enhancing the quality of life for both patients and their dedicated caregivers.

1.1 Purpose

The purpose of Thought Locker is to provide assistance to individuals with mild to moderate dementia in their daily tasks. This innovative solution offers various features such as reminders, item location, and contact support, aiming to support individuals in their daily routines. Additionally, it allows caregivers to closely monitor the progress and mood changes of those under their care. Thought Locker is designed to be accessible on both Android and iOS devices, with a focus on user-friendly authentication for ease of use. It also encompasses a calendar, item finders, and location tracking sensors. Caregivers have access to motion sensors and can monitor patient activities effectively. Furthermore, the system is highly customizable, tailoring its features to meet the unique needs of each patient, while still allowing some degree of control for the patients themselves. This comprehensive solution combines hardware, in the form of a mobile device, with software components like JavaScript, React, and Node.js. It uses GitHub for version control, issue tracking, and continuous integration, ensuring a robust and reliable platform for both patients and caregivers.
1.2 Scope

The purpose of Thought Locker is to provide support for dementia patients, families, friends, and caregivers, with a primary focus on promoting independence and minimizing stress. The project's scope is demonstrated through its prototype, which highlights essential functionalities, although not encompassing all the features expected in a real-world scenario. To achieve this, certain hardware and software requirements are necessary, such as Android compatibility, JavaScript, React, Node.js, MySQL, AWS, Jest, Git/GitLab, and more. The prototype features a range of elements, including user accounts, task management, calendars, item tracking, and notifications. Some of these features employ simulated data, specifically for item location and contact information. The development process presents challenges that extend to aspects like becoming familiar with design tools, navigating the intricacies of React Native, effectively using GitLab, integrating databases, setting up servers, interfacing with APIs, and sourcing simulated data.

[ This space left intentionally blank. ]
1.3 Definitions, Acronyms, and Abbreviations

**Amazon Web Services (AWS):** A cloud computing platform that provides a variety of services including compute, storage, databases, analytics, machine learning, networking, mobile, developer tools, security, and enterprise applications.

**Android:** An open-source mobile operating system based on the Linux kernel and developed by Google.

**Application Programming Interface (API):** A set of protocols, routines, and tools for building software applications that specify how software components should interact with each other.

**Application Server:** A software framework that provides an environment for running applications.

**Authentication:** The process of verifying the identity of a user.

**Biometric:** Unique physical or behavioral characteristics of an individual to identify them.

**Bluetooth Low Energy (BLE):** A wireless communication technology used for short-range communication between devices.

**Caregiver/Family Member:** Any person that is able to provide assistance with managing a dementia patient’s symptoms. Their duties consist of transporting patients to and from their commitments, maintaining a patient’s daily routine, and reminding them to take their medications at regular intervals.

**Continuous Integration (CI):** A software development practice that involves frequently integrating code changes from multiple developers into a shared repository, verifying that the changes do not break the build and that the software continues to function correctly.
**Containerization**: A method of packaging and deploying software applications with all their dependencies into a single unit, called a container, which can run reliably and consistently across different computing environments.

**Database Server**: A computer program or software application that provides database services to other computer programs or clients.

**Docker**: A software platform that allows its users to build, test, and deploy applications in standardized executables resembling containers.

**GeoTag**: A physical tag that emits a Bluetooth signal to assist devices in determining their location.

**GitLab**: An open-source repository service that allows its users to work on a single project simultaneously.

**Google Maps Geolocation API**: a service provided by Google that allows developers to determine the location of a device using Wi-Fi or mobile network signals.

**iOS**: A mobile operating system developed by Apple Inc based on the Unix operating system.

**Issue Tracking**: The process of managing and resolving software issues, bugs, and feature requests.

**JavaScript**: Scripting programming language that creates dynamic web page content and mobile applications.

**JavaScript Object Notation (JSON)**: A lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate.

**Jest**: Test-runner for JavaScript applications that supports a JavaScript library for creating, running, and structuring tests.
**Jira**: A project management and issue tracking tool developed by Atlassian. It is widely used by software development and project management teams to plan, track, and manage their work.

**MongoDB**: Non-relational document database that provides support for non-relational querying.

**Node.js**: An open-source, cross-platform JavaScript runtime environment that enables the execution of server-side JavaScript code.

**Non-relational Database**: A type of database that doesn't rely on the traditional structure of tables, columns, and rows found in relational databases. Instead, they are designed to handle large and complex sets of unstructured, semi-structured, or structured data.

**Patient**: Any individual suffering from mild to moderate dementia. Their symptoms typically consist of occasional disorientation, difficulties with making decisions, and short-term memory loss.

**PostgreSQL**: Free and open-source database management system that supports relational (SQL) and non-relational (JSON) querying.

**Programming Language**: A formal language used to communicate instructions to a computer or other machine.

**React**: An open-source JavaScript library that is used to build user interfaces for web and mobile applications.

**Relational Database**: A type of database that stores data in a structured format, using rows and columns to represent data entities and attributes.

**Relational Database Management System (RDBMS)**: A software system that enables users to create, maintain, and manipulate relational databases.

**Repository**: A central location where digital files, usually in the form of software source code, are stored and managed.
Structured Query Language (SQL): A programming language used to manage and manipulate data in a relational database management system (RDBMS).

Testing Framework: A set of guidelines, standards, and tools that software developers use to create and run automated tests for their code.

Version Control: A system that tracks and manages changes to software code, documentation, and other files over time.

[ This space intentionally left blank. ]
1.4 References


clock&hl=en_US&gl=US&pli=1


1.5 Overview

In this overview, the product specification is set to encompass the design of Thought Locker's architecture and deliver a functional description of the prototype. This architecture encompasses the necessary hardware for accessing Thought Locker along with the development software. The software utilized within Thought Locker includes all the essential programming languages for both the front-end and back-end components. Furthermore, the detailed documentation will outline the functional requirements of Thought Locker and how they are implemented within the prototype.

2 General Description

The Thought Locker prototype is a mobile application consisting of a set of features to assist those with dementia and caregivers.

2.1 Prototype Architecture Description

The Prototype Architecture Description outlines two primary user categories: Patients and Caregivers. The prototype application is intended to run on both Android and iOS platforms. For server hosting, The Thought Locker has chosen AWS as the preferred service provider for our Web Application. The front-end and back-end components will be seamlessly interconnected through the React Native API. In terms of the database, the Thought Locker prototype will utilize MYSQL, serving as the repository for storing vital end user information, profiles, and historical data. Additionally, React Native APIs will facilitate the connection between the ThoughtLocker's database and its server, while the prototype will also make use of the Google Maps API for location generation.
2.2 Prototype Functional Description

Referring to Figure 1, Thought Locker prototype's major functional components encompass various essential features. To begin with, there is a User Registration system in place that ensures a secure and distinct connection between patients and caregivers. This process involves the creation of user profiles that are seamlessly linked to the database. Following that, the Login Authentication mechanism allows for the secure access to the application by utilizing the username and password established during the registration process. Moreover, the User Profile Management function empowers users to update their profiles, delete them, and modify permissions as needed. The application also provides a Calendar feature, enabling users to view their schedules. Task Scheduling is another critical component, allowing users to access and track their assigned tasks, with the option to delete or modify them if authorized. For efficient item tracking, the Item Locator feature utilizes Bluetooth technology and audio cues, supported by visual aids. In addition, the Contact Center functionality enables users to reach
pre-programmed contacts swiftly. In times of urgency, the Emergency Contacts function lets users quickly cycle through a predefined list of emergency contacts with a single button press. The system offers monitoring capabilities, enabling users to configure and receive alerts from pre-placed sensors. These notifications are also relayed to caregivers, facilitating timely responses. The Event Viewer allows users to access application logs, primarily serving the Caregiver role in reviewing the records. Lastly, the User Analytics feature provides users with valuable insights and the ability to download data related to logins, task management, item tracking, contact logs, and monitoring.

**Table 1**

*Thought Locker Feature Description and Prototype Implementation*

<table>
<thead>
<tr>
<th>Features</th>
<th>Description</th>
<th>Prototype Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ACCOUNT MANAGEMENT</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Account Creation</td>
<td>Set up a new patient or caregiver account</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Account Deletion Verification</td>
<td>Ensure the intentional removal of a user's account, requiring confirmation steps to prevent accidental deletion</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Login / Authentication</td>
<td>Users prove their identity and gain access to the application by providing the required credentials</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>User Profile Management</td>
<td>Administration of individual user accounts within the application, allowing users to customize their settings, preferences, and personal information</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>User Location Information</td>
<td>Identifies the geographic position of a user through their device by enabling location-based</td>
<td>Partially Implemented: Location data will be simulated</td>
</tr>
<tr>
<td>Service</td>
<td>Description</td>
<td>Implementation Status</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td><strong>USER INTERFACE</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task Scheduling</td>
<td>Organize specific tasks to be executed at predetermined times</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Task Completion</td>
<td>Mark a task complete after successfully executing it</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Task Deletion Verification</td>
<td>Ensure the intentional removal of a specific task, requiring confirmation steps to prevent accidental deletion</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Item Locator</td>
<td>Identifies the location of items, making it easier to find and manage them</td>
<td>Partially Implemented: Location data will be simulated</td>
</tr>
<tr>
<td>Item Registration</td>
<td>Record and document information about an item in a database, including details such as ownership, specifications, and relevant data for tracking and management</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Item Deletion Verification</td>
<td>Ensure the intentional removal of a specific item from the database, requiring confirmation steps to prevent accidental deletion</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Event Viewer</td>
<td>Provides logs and details about system, security, and application events, aiding in system troubleshooting, monitoring, and maintenance</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Calendar</td>
<td>Visual tool that organizes and displays dates, days, and events, facilitating time management and scheduling</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Sensor Monitoring</td>
<td>Continuous observation and collection of data from sensors to track and assess changes to item</td>
<td>Partially Implemented: Sensor data will be simulated</td>
</tr>
<tr>
<td>Emergency Contacts</td>
<td>Individuals designated to be contacted in urgent or critical situations, typically for medical, safety, or personal reasons</td>
<td>Partially Implemented: Contact data will be simulated</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>Notifications</td>
<td>Alerts delivered through the application to inform users about important information, events, or updates</td>
<td>Fully Implemented</td>
</tr>
</tbody>
</table>

**DATA MANAGEMENT**

<table>
<thead>
<tr>
<th>User Analytics</th>
<th>Collecting and analyzing data about the behavior and interactions of patients with the application, aiding in decision-making and patient care</th>
<th>Partially Implemented: Analytics will be simulated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reports</td>
<td>Structured documents that provide information, analysis, and findings on specific user defined topics</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Item Tagging</td>
<td>Assign key metadata values to items, enabling efficient categorization and retrieval within the database</td>
<td>Partially Implemented: Geotag data will be simulated</td>
</tr>
<tr>
<td>Images</td>
<td>Upload images of tracked items, which must correspond to items in the database</td>
<td>Fully Implemented</td>
</tr>
</tbody>
</table>

[ This space left intentionally blank. ]
2.3 External Interfaces

2.3.1 Hardware Interfaces

In the context of external interfaces, our system requires specific hardware compatibility. This includes an Android mobile device running Android 11 or a more recent version, an iOS mobile device operating on iOS 15 or a more recent iteration, and a computer running Linux, Windows, or Mac OS equipped with a contemporary web browser. These hardware interfaces serve as the essential gateways for our system to function seamlessly and provide users with a versatile experience.

2.3.2 Software Interfaces

The software interfaces in question encompass a variety of tools and platforms. GitLab serves as a crucial component for version control and collaborative software development. MySQL is employed as a relational database management system, handling data storage and retrieval. AWS, Amazon's cloud computing platform, is utilized for hosting and scaling applications, ensuring robust performance and availability. Android Studio, the integrated development environment, facilitates the creation and management of Android applications. These interfaces collectively support various aspects of software development and deployment.

2.3.3 User interfaces

In the realm of user interfaces, a pivotal aspect to consider is the integration of mobile devices equipped with internet connectivity. These devices play a significant role in shaping user interactions with various applications and services. Mobile interfaces, given their ubiquitous access to the internet, have the potential to enhance the user experience by providing easy and versatile access to a multitude of online resources. This
technology has become an integral part of modern life, enabling users to seamlessly interact with digital platforms and access a wide range of information and services through the power of internet connectivity. In sum, mobile devices with internet access stand as a vital element in the landscape of user interfaces, greatly influencing the ways in which individuals engage with digital content and applications.

2.3.4 Communications Protocols and Interfaces

Within communications protocols and interfaces, IEEE 802.11b-ax is a pivotal standard in the IEEE 802.11 family, advancing wireless technology for enhanced connectivity with improved data rates, network capacity, and performance across various applications and industries.